ECSE426 - Microprocessor Systems
Group 20 - Lab 3 & 4

Fabrice Normandin 260636800
Marcel Morin 260605670

March 19, 2018

Abstract

This report outlines the engineering process we used while completing Labs 3 & 4. These most recent labs built upon the ADC sampling, filtering and 7-segment display work done previously as part of Lab 2. The main objective of this lab was to create a system which accepts user input through a keypad, and then attempts to match the given value (a RMS voltage) by using a PWM signal. This system also had to successfully manage the transitions between the different states of operation (sleep, keypad input, voltage matching), all-the-while conserving CPU resources as much as possible. The system was first created using an interrupt-driven approach (Lab3), and later using the multithreaded paradigms of the CMSIS_RTOS framework (Lab4). Despite some portions of this lab presenting a significant technical challenge, it can be considered a resounding success, as all of its objectives were successfully completed. While our current solution makes a great effort towards CPU efficiency, additional measures could be taken in order to reduce the power consumption even further.

Todo list

Explain the PWM timer (lots of theory there) 4
Explain Threads/multithreading ? 4
Explain Interrupt Service Routines ? 4

Contents

1 Problem Statement **3**

2 Theory and Hypothesis **4**

3 Implementation **5**
3.1 High-Level System Architecture 5
3.2 Display Thread 5
3.3 Keypad Thread 5
3.4 ADC Logic 5
3.5 PWM Timer 5

4 Testing and Observations **6**

5 conclusion **7**

1 Problem Statement

A comprehensive list of all system requirements can be found within the Lab 3 & Lab 4 handouts. Among these requirements, one of the most challenging to meet was one related to the keypad, as it showed that the system had to respond differently to buttons depending on their press duration, and generate a transition between different modes of operation (a long “” press would put the system into sleep mode, for example). Consequently, an additional requirement was created, by which the system code had to be structured as a finite-state machine (FSM), and successfully manage state transitions in a clear, well-structured way. State diagrams were also deemed essential, and we considered it a system requirement that each sub-system have its own related state diagram, in order to clearly represent the functioning of our system.

2 Theory and Hypothesis

Explain the PWM timer (lots of theory there)

Explain Threads/multithreading ?

Explain Interrupt Service Routines ?

3 Implementation

This section will describe the design and engineering processes that were used while implementing our system, based on the requirements described above. In order to simplify and structure its description, the system will first be broken down into its different logical components.

3.1 High-Level System Architecture

The various requirements and corresponding program functionality of our system can be broken down into four main areas: the 7-segment display, the keypad, the ADC and the PWM timer. Despite the overall system having significant complexity, there is a very limited number of states for the overall system, with the transition between them being very well defined. A state diagram showing the three states, along with the name of their transitions can be seen in Figure 3.1.

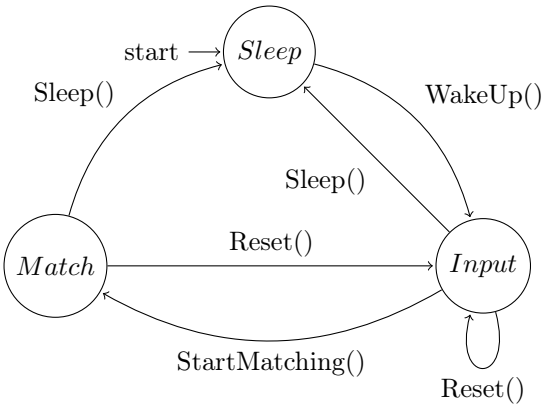
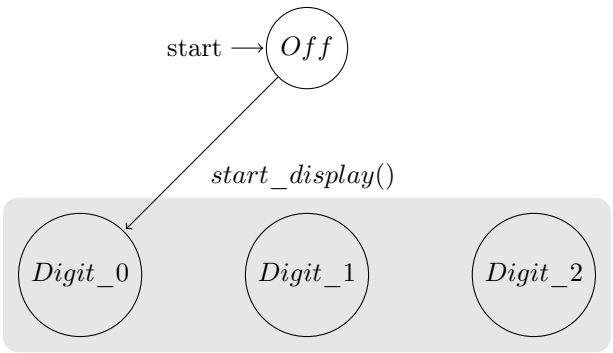


Figure 1: System high-level state diagram

Sleep	Input	Match
The sleep state (blabla)	The Input state (blabla)	The Match state is (blabla)

3.2 Display Thread



3.3 Keypad Thread

3.4 ADC Logic

3.5 PWM Timer

4 Testing and Observations

5 conclusion